**Packet Transfer Protocol**

(Rev 1.0)

**Packet Format**

| START BYTE | CMD | Payload Length | <Payload> | <CRC16> | END BYTE |

- START BYTE: 0xAA

- END BYTE: 0xBB

- CMD: Command code indicating the packet type

- Payload Length: Length of the payload in bytes (excluding START BYTE, CMD, CRC16, and END BYTE)

- Payload: Data associated with the command (varies based on CMD type)

- CRC16: 16-bit CRC checksum for packet integrity verification

**Packet Descriptions**

1. Start/Header Packet

| START BYTE | CMD | Payload Length | <Payload> | <CRC16> | END BYTE |

- CMD: 0x01 (Start/Header Packet)

- Payload:

- File Length (4 bytes, represents total file size)

- File Checksum (4 bytes, 32-bit checksum of the file)

- Version Count (2 bytes, version number of the file)

- Payload Length: 10 bytes (total of file length, checksum, and version count)

2. Erase Memory Packet

| START BYTE | CMD | Payload Length | <Payload> | <CRC16> | END BYTE |

- CMD: 0x02 (Erase Memory Command)

- Payload: No payload

- Payload Length: 0 (indicates no additional data)

3. Firmware Packet

| START BYTE | CMD | Payload Length | <Payload> | <CRC16> | END BYTE |

- CMD: 0x03 (Firmware Data Packet)

- Payload:

- Firmware Data bytes (actual firmware chunk, up to the specified Payload Length)

- Payload Length: Length of the firmware data in the current packet

4. End Packet

| START BYTE | CMD | Payload Length | <Payload> | <CRC16> | END BYTE |

- CMD: 0x04 (End of Transfer)

- Payload: No payload

- Payload Length: 0 (indicates no additional data)

5. Update Firmware Packet

| START BYTE | CMD | Payload Length | <CRC16> | END BYTE |

- CMD: 0x05 (Update Firmware Command)

- Payload: No payload

- Payload Length: 0 (indicates no additional data)

6. Response Packet

| START BYTE | CMD | Payload Length | Response Code | <CRC16> | END BYTE |

- CMD: Response to the previous command

- Response Code:

- 0: OK

- 1: NACK

- 2: Checksum Verification Failed

- 3: Firmware Update Failed

- Payload Length: 1 (length of the response code)

**CRC16 Calculation**

To ensure packet integrity, a CRC16 checksum is calculated for the CMD, Payload Length, and Payload fields. Below is the implementation of the CRC16 calculation used

|  |  |
| --- | --- |
| * Initial CRC Value: The initial CRC value is set to 0xFFFF. * Polynomial: The polynomial used for the CRC calculation is 0x2024. * The CRC is calculated over the CMD, Payload Length, and Payload fields, excluding the START BYTE and END BYTE. * The <CRC16> value should be appended before sending the packet, and verified upon reception. |  |

* Initial CRC Value: The initial CRC value is set to 0xFFFF.
* Polynomial: The polynomial used for the CRC calculation is 0x2024.
* The CRC is calculated over the CMD, Payload Length, and Payload fields, excluding the START BYTE and END BYTE.
* The <CRC16> value should be appended before sending the packet, and verified upon reception.

**Example Packet Structures**

1. **Start/Header Packet Example**:

0xAA 0x01 0x0A 0x00 0x00 0x02 0x58 0x12 0x34 0x56 0x78 0x01 0x00 <CRC16> 0xBB

**Explanation**:

* 0xAA: START BYTE
* 0x01: CMD: Start/Header
* 0x0A: Payload Length: 10 bytes
* 0x00 0x00 0x02 0x58: File Length (600 bytes)
* 0x12 0x34 0x56 0x78: File Checksum
* 0x01 0x00: Version Count
* <CRC16>: 16-bit CRC value calculated for the above fields
* 0xBB: END BYTE

1. **Response Packet Example (OK Response)**:

0xAA 0x01 0x01 0x00 <CRC16> 0xBB

0xAA: START BYTE

0x01: CMD: Response to Start/Header

0x01: Payload Length: 1 byte (Response Code)

0x00: Response Code: OK

<CRC16>: 16-bit CRC value for the above fields

0xBB: END BYTE

**Troubleshooting**

| **Error Type** | **Possible Causes** | **Suggested Action** |
| --- | --- | --- |
| **Checksum Verification Failed** | Incorrect CRC16 calculation, data corruption | Recalculate CRC or request retransmission |
| **Firmware Update Failed** | File integrity issues, version mismatch, memory error | Retry update, check memory health |
| **NACK Received** | Invalid command or sequence | Ensure the correct command sequence |